



# JSP Quick Reference Card

## Basic Syntax

### Default scripting language

The scripting language of a JSP page defaults to Java. Insert the following line in a JSP page to configure the page to use JavaScript:

```
<%@ page language = "javascript" %>
```

### Using white space

White space contained within the template code is returned to the client as it was entered in the JSP.

### Quoting attribute values

Quote attribute values, using either single or double quotes, to all JSP elements. For example:

```
<%@ page contentType = "text/plain" %>
```

### Writing comments for the JSP

A JSP comment is not output to the client as part of the JSP page's output.

```
<!-- Comment string... -->
```

### Outputting comments to the client

HTML comments are output to the client.

```
<!-- comments -->
```

## Directives

### page

Defines page-wide attributes.

```
<%@ page attribute="value" ... %>
```

Attributes, with default values, are:

```
attribute = language="java" | session="true"
| contentType=text/html; charset="ISO-8859-1"
| import="package(s)" | buffer="8kb"
| autoFlush="true" | isThreadSafe="true"
| info="text_string" | errorPage="relativeURL"
| isErrorPage="true" | extends="class_name"
```

value = a string literal in single or double quotes.

### include

Inserts text into a JSP page.

```
<%@ include file = "path" ... %>
```

### taglib

Defines a custom tag library used by a JSP page.

```
<%@ taglib uri="tagLibraryURI"
prefix="tagPrefix" %>
```

After the taglib directive, reference the custom tags using the syntax:

```
<tagPrefix: tagName>
...
</tagPrefix: tagName>
```

## Scripting Elements

### declaration

Creates page-wide definitions such as variables.

```
<%! declaration %>
```

Example:

```
<%! private String foo = null;
public String getFoo() {return this.foo;} %>
```

## scriptlet

Contains a block of scripting code. A JSP page can contain multiple blocks of scripting code.

```
<% script code %>
```

Example:

```
<% String greeting =
request.getParameter("Greeting");
out.println(greeting); %>
```

## expression

Defines statements evaluated on the server before sending the page output to the client.

```
<%= expression %>
```

Example:

```
<%= myVar1%>
```

## Actions

### jsp:include

Call one JSP page from another. Upon completion, the destination page returns control to the calling page.

```
<jsp:include page="path" flush="true" />
```

```
<jsp:include page="path" flush="true">
<jsp:param name="paramName"
value="paramValue" /> ...
</jsp:include>
```

### jsp:forward

Calls one JSP page from another. Execution of the calling page is terminated by the call.

```
<jsp:forward page="path" />
```

```
<jsp:forward page="path">
<jsp:param name="paramName"
value="paramValue" /> ...
</jsp:forward>
```

## jsp:plugin

Enables you to invoke an applet on a client browser.

```
<jsp:plugin
  type="bean|applet"
  code="objectCode"
  codebase="objectCodebase"
  { alignment="alignment" }
  { archive="archiveList" }
  { height="height" }
  { hspace="hspace" }
  { jreversion="jreversion" }
  { name="componentName" }
  { vspace="vspace" }
  { width="width" }
  { nspuginurl="url" }
  { ipluginurl="url" } >
  { <jsp:params>
    { <jsp:param name="paramName"
      value="paramValue" /> }+
  }
</jsp:plugin>
```

The elements in brackets ({}) are optional.

## jsp:useBean

Defines an instance of a Java bean.

```
<jsp:useBean id="name"
  scope="page|request|session|application"
  typeSpec />

<jsp:useBean id="name"
  scope="page|request|session|application"
  typeSpec >
  body
</jsp:useBean>
```

*typespec* is any one of the following:

```
class="className" |
class="className" type="typeName" |
beanName="beanName" type="typeName" |
type="typeName"
```

## jsp:setProperty

Sets the value of one or more properties in a bean.

```
<jsp:setProperty name="beanName" prop_expr />

prop_expr has one of the following forms:

property="*" |
property="propertyName" |
property="propertyName" param="parameterName" |
property="propertyName" value="propertyValue"
```

## jsp:getProperty

Writes the value of a bean property as a string to the out object.

```
<jsp:getProperty name="name"
  property="propertyName" />
```

## JSP Objects

See the corresponding Java object type for the available methods for these objects.

### application

The servlet context obtained from the servlet configuration object.

**Java type:** `javax.servlet.ServletContext`

### config

The `ServletConfig` object for the JSP page.

**Java type:** `javax.servlet.ServletConfig`

### exception

The uncaught exception that resulted in the error page being invoked.

**Java type:** `java.lang.Throwable`

### out

An object that writes into a JSP page's output stream.

**Java type:** `javax.servlet.jsp.JspWriter`

## pageContext

The page context for the JSP.

**Java type:** `javax.servlet.jsp.PageContext`

## request

The client request.

**Java type:** `javax.servlet.HttpServletRequest`

## response

The response to the client.

**Java type:** `javax.servlet.HttpServletResponse`

## session

The session object created for the requesting client.

**Java type:** `javax.servlet.http.HttpSession`

## Allaire Contact Information

### Allaire Web sites

Main Allaire Web site:

[www.allaire.com](http://www.allaire.com)

JRun Development Center:

[www.allaire.com/development/jrunreferencedesk/](http://www.allaire.com/development/jrunreferencedesk/)

JRun Developer Forum:

[forums.allaire.com/jrunconf](http://forums.allaire.com/jrunconf)

### Allaire technical support

Telephone support is available Monday through from Friday 8 AM to 8 PM Eastern time (except holidays).

Toll Free: 888.939.2545 (U.S. and Canada)

Telephone: 617.761.2100 (outside U.S. and Canada)

JRun is a trademark of Allaire Corporation. All other trademarks are property of their respective holder(s.)  
© 2000 Allaire Corporation. All rights reserved.  
Part number: AA-JRQRF-RK